UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/765,145	01/28/2004	Eun Hye Choi	248156US2RD	9722

22850 7590 01/10/2008
OBLON, SPIVAK, MCCLELLAND MAIER & NEUSTADT, P.C.
1940 DUKE STREET
ALEXANDRIA, VA 22314

EXAMINER	
LE, MIRANDA	

ART UNIT	PAPER NUMBER
2167	

NOTIFICATION DATE	DELIVERY MODE
01/10/2008	ELECTRONIC

Please find below and/or attached an Office communication concerning this application or proceeding.

The time period for reply, if any, is set in the attached communication.

Notice of the Office communication was sent electronically on above-indicated "Notification Date" to the following e-mail address(es):

patentdocket@oblon.com
oblonpat@oblon.com
jgardner@oblon.com

Office Action Summary	Application No. 10/765,145	Applicant(s) CHOI ET AL.	
	Examiner Miranda Le	Art Unit 2167	

-- *The MAILING DATE of this communication appears on the cover sheet with the correspondence address* --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on 22 October 2007.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) 1-21 is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) 1-21 is/are rejected.

7)☐ Claim(s) _____ is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☒ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____ .
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

Continued Examination Under 37 CFR 1.114

1. A request for continued examination under 37 CFR 1.114, including the fee set forth in 37 CFR 1.17(e), was filed in this application after final rejection. Since this application is eligible for continued examination under 37 CFR 1.114, and the fee set forth in 37 CFR 1.17(e) has been timely paid, the finality of the previous Office action has been withdrawn pursuant to 37 CFR 1.114. Applicant's submission filed on 10/22/2007 has been entered.

2. This communication is responsive to Amendment, filed 10/22/2007.

Claims 1-21 are pending in this application. Claims 1, 19, 20, 21 are independent claims. In the Amendment, claims 1, 19, 20 has been amended, claim 21 has been added. This action is made non-Final.

3. The rejection of claim 20 under 35 U.S.C. §101 has been withdrawn in view of the amendment.

Information Disclosure Statement

4. Applicants' Information Disclosure Statement, filed 01/03/2008, has been received, entered into the record, and considered. See attached form PTO-1449.

Claim Rejections - 35 USC § 101

5. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

6. Claim 19 is rejected under 35 U.S.C. § 101 because the claimed invention is directed to non-statutory subject matter.

Claim 19 recites "a transactional processing system..., comprising: a copying unit...; a judging unit; a processing unit...; a reflecting unit..."; however, each recited element of the system may be reasonably interpreted by one of ordinary skill as software alone; for example, par [0294] indicates "the transaction processing system of each of the above described embodiments can be conveniently implemented in a form of a software package" suggesting the claim as a whole can be implemented using software means only, as these units that make up the system are all software applications that do not result in a tangible practical application under 35 U.S.C. § 101; thus, the system is not tangible embodied in a manner so as to be executable. The claim lacks the necessary physical articles or objects to constitute a machine or a manufacture within the meaning of 35 U.S.C. § 101, instead being software per se.

As such, the claimed system does not define any specific hardware and needs to be amended to include physical computer hardware (e.g. processor, memory) to execute the software components. See MPEP 2106.01.

Claim Rejections - 35 USC § 103

7. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

This application currently names joint inventors. In considering patentability of the claims under 35 U.S.C. 103(a), the examiner presumes that the subject matter of the various claims was commonly owned at the time any inventions covered therein were made absent any evidence to the contrary. Applicant is advised of the obligation under 37 CFR 1.56 to point out the inventor and invention dates of each claim that was not commonly owned at the time a later invention was made in order for the examiner to consider the applicability of 35 U.S.C. 103(c) and potential 35 U.S.C. 102(e), (f) or (g) prior art under 35 U.S.C. 103(a).

8. Claims 1-6, 8, 9, 15, 19, 20 are rejected under 35 U.S.C. 103(a) as being unpatentable over Weedon et al. (US Patent No. 6,732,239), in view of Schrader et al. (US Patent No. 5,903,881).

As per claim 1, Weedon teaches a concurrency control method in a transaction processing system for processing a plurality of transactions in parallel with respect to data, the concurrency control method comprising:

producing a copy *(i.e. The invention permits users to enjoy both a performance benefit from the exclusive mode caching as well as a concurrency benefit of having multiple copies of the object accessible at one time, col. 3, lines 1-11)* of the data at a time of starting an access to the data by each transaction *(i.e. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary)*;

judging whether *(i.e. if the test at step 280 determines that the counter is zero, then the just completed transaction is the last outstanding transaction. In that case, a test is made at step 235 to determine whether the transaction is to be committed to the database. As described above, if the transaction is to be rolled back, then the process flow ends at step 240. Otherwise, the updates are committed to the data store by copying the record in that transaction to the data store, as shown at step 245, and by putting the contents of that object in the exclusive mode cache 150, as indicated at step 250, col. 5, lines 57-67)* a collision between one of reading access or writing access *(i.e. completing any current transactions by selectively committing any updates that were made to the object with a write operation to the data store, Summary)* to be made by a first transaction with respect to a copy of the data for the first transaction *(i.e. responding to a first transaction by retrieving a cache version of the object from the exclusive access cache, Summary)* and another one of the reading access or writing access made by the second transaction with respect to a copy of the data for the second transaction will occur or not *(i.e. the last transaction being either the first transaction or one of the concurrent transactions, if any, Summary)* when the first transaction and the second transaction are accepted at the same time as concurrent transaction for accessing the same location of the data *(i.e. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary)*;

carrying out a processing *(i.e. If the counter is not zero, that indicates that concurrent transactions are being processed. Updates of the just-completed transaction (from step 260), if any, may have to be copied to the data store. A test is made at step 285 to determine whether there is any non-conflicting update to commit, and if there is, the data store is updated at step*

290. In this circumstance, multiple copies of the data are out for processing, with the first transaction having exclusive access to the exclusive mode cache, and subsequent concurrent transactions having access to copies of the data from the data store. In sharp contrast to the prior art, these transactions can be updated to the data store while the exclusive mode cache is locked up for the first transaction, col. 5, lines 44-56) for avoiding the collision due to the concurrent transactions when the judging step judges that the collision will occur *(i.e. if the test at step 280 determines that the counter is zero, then the just completed transaction is the last outstanding transaction. In that case, a test is made at step 235 to determine whether the transaction is to be committed to the database. As described above, if the transaction is to be rolled back, then the process flow ends at step 240. Otherwise, the updates are committed to the data store by copying the record in that transaction to the data store, as shown at step 245, and by putting the contents of that object in the exclusive mode cache 150, as indicated at step 250, col. 5, lines 57-67)*;

and reflecting a writing access made by the first transaction with respect to a copy of the data for the first transaction *(i.e. As part of the completion process, the sharing counter is decremented. If the value after decrementing is zero, that indicates that the transaction currently completing is the only transaction to have a copy of the record at that point in time. If the post-decrement counter value is zero, and if the transaction is committing (meaning that the record is being copied back to the database) then this record is also put back into the exclusive mode cache, to be used by subsequent transactions, col. 6, lines 48-56)*, on the data, when the first transaction is to be finished normally, and reflecting the writing access also on a copy of the data for the second transaction if the second transaction is not finished yet *(i.e. Each transaction is*

subsequently completed, meaning that it is either committed or rolled back. If the transaction is committed, it means that the copy of the record is written back to the database (if it was modified). If the transaction was rolled back, it means that the copy of the record is thrown away, col. 6, lines 41-47).

Weedon does not explicitly teach hierarchical data limitation.

Schrader teaches this limitation *(i.e. The database module 1407 is an interface to the user's data, which is stored in the transaction database. Records may be fetched or saved by the database module 1407 and this module allows all the other modules to access the transaction database. The database module 1403 preferably stores the user's data in combined relational-hierarchical data model, though other data models may also be employed. A hierarchical model is used to organize accounts per financial institution, such that all transactions for each account are stored with the account. A relational model is used for individual transactions, to provide associations between payees, amounts, and individual transactions within each account. Data that is stored includes payment data, transactions data, funds transfer data, and e-mail data. Transaction records may be variable length. Each record in the transaction data is marked as being either in the online statement 150, the mini checkbook 181, or the out box 167, and as being either reconciled or unreconciled. This marking allows for subsequent auto-reconciliation and reporting. Some transactions in the online statement 150 may be unreconciled if the user did not enter them first in the mini checkbook 181. col. 13, line 65 to col. 14, line 19).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon and Schrader at the time the invention was made to modify the system of Weedon to include the hierarchical data as taught by Schrader. One of ordinary skill in the art would be

motivated to make this combination in order to store the user's data in combined relational-hierarchical data model in view of Schrader (col. 13, line 14 to col. 14, line 19), as doing so would give the added benefit of obtaining a hierarchical model to organize accounts per financial institution, such that all transactions for each account are stored with the account as taught by Schrader (col. 13, line 65 to col. 14, line 19).

As per claim 19, Weedon teaches a transaction processing system for processing a plurality of transactions in parallel with respect to data, comprising:

a copying unit configured to produce a copy *(i.e. The invention permits users to enjoy both a performance benefit from the exclusive mode caching as well as a concurrency benefit of having multiple copies of the object accessible at one time, col. 3, lines 1-11)* of the data at a time of starting an access to the data by each transaction *(i.e. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary)*;

judging unit configured to judge whether *(i.e. if the test at step 280 determines that the counter is zero, then the just completed transaction is the last outstanding transaction. In that case, a test is made at step 235 to determine whether the transaction is to be committed to the database. As described above, if the transaction is to be rolled back, then the process flow ends at step 240. Otherwise, the updates are committed to the data store by copying the record in that transaction to the data store, as shown at step 245, and by putting the contents of that object in the exclusive mode cache 150, as indicated at step 250, col. 5, lines 57-67)* a collision between one of reading access or writing access *(i.e. completing any current transactions by selectively committing any updates that were made to the object with a write operation to the data store,*

Summary) to be made by a first transaction with respect to a copy of the data for the first transaction *(i.e. responding to a first transaction by retrieving a cache version of the object from the exclusive access cache, Summary)* and another one of the reading access or writing access made by the second transaction with respect to a copy of the data for the second transaction will occur or not *(i.e. the last transaction being either the first transaction or one of the concurrent transactions, if any, Summary)* when the first transaction and the second transaction are accepted at the same time as concurrent transaction for accessing the same location of the data *(i.e. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary)*;

a processing unit configured to carry out a processing *(i.e. If the counter is not zero, that indicates that concurrent transactions are being processed. Updates of the just-completed transaction (from step 260), if any, may have to be copied to the data store. A test is made at step 285 to determine whether there is any non-conflicting update to commit, and if there is, the data store is updated at step 290. In this circumstance, multiple copies of the data are out for processing, with the first transaction having exclusive access to the exclusive mode cache, and subsequent concurrent transactions having access to copies of the data from the data store. In sharp contrast to the prior art, these transactions can be updated to the data store while the exclusive mode cache is locked up for the first transaction, col. 5, lines 44-56)* for avoiding the collision due to the concurrent transactions when the judging step judges that the collision will occur *(i.e. if the test at step 280 determines that the counter is zero, then the just completed transaction is the last outstanding transaction. In that case, a test is made at step 235 to determine whether the transaction is to be committed to the database. As described above, if the*

transaction is to be rolled back, then the process flow ends at step 240. Otherwise, the updates are committed to the data store by copying the record in that transaction to the data store, as shown at step 245, and by putting the contents of that object in the exclusive mode cache 150, as indicated at step 250, col. 5, lines 57-67);

and reflecting unit configured to reflect a writing access made by the first transaction with respect to a copy of the data for the first transaction *(i.e. As part of the completion process, the sharing counter is decremented. If the value after decrementing is zero, that indicates that the transaction currently completing is the only transaction to have a copy of the record at that point in time. If the post-decrement counter value is zero, and if the transaction is committing (meaning that the record is being copied back to the database) then this record is also put back into the exclusive mode cache, to be used by subsequent transactions, col. 6, lines 48-56)*, on the data, when the first transaction is to be finished normally, and reflecting the writing access also on a copy of the data for the second transaction if the second transaction is not finished yet *(i.e. Each transaction is subsequently completed, meaning that it is either committed or rolled back. If the transaction is committed, it means that the copy of the record is written back to the database (if it was modified). If the transaction was rolled back, it means that the copy of the record is thrown away, col. 6, lines 41-47).*

Weedon does not specifically teach hierarchical data limitation.

Schrader teaches this limitation *(i.e. The database module 1407 is an interface to the user's data, which is stored in the transaction database. Records may be fetched or saved by the database module 1407 and this module allows all the other modules to access the transaction database. The database module 1403 preferably stores the user's data in combined relational-*

hierarchical data model, though other data models may also be employed. A hierarchical model is used to organize accounts per financial institution, such that all transactions for each account are stored with the account. A relational model is used for individual transactions, to provide associations between payees, amounts, and individual transactions within each account. Data that is stored includes payment data, transactions data, funds transfer data, and e-mail data. Transaction records may be variable length. Each record in the transaction data is marked as being either in the online statement 150, the mini checkbook 181, or the out box 167, and as being either reconciled or unreconciled. This marking allows for subsequent auto-reconciliation and reporting. Some transactions in the online statement 150 may be unreconciled if the user did not enter them first in the mini checkbook 181, col. 13, line 65 to col. 14, line 19).

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon and Schrader at the time the invention was made to modify the system of Weedon to include the hierarchical data as taught by Schrader. One of ordinary skill in the art would be motivated to make this combination in order to store the user's data in combined relational-hierarchical data model in view of Schrader (col. 13, line 14 to col. 14, line 19), as doing so would give the added benefit of a hierarchical model is used to organize accounts per financial institution, such that all transactions for each account are stored with the account as taught by Schrader (col. 13, line 65 to col. 14, line 19).

As per claim 20, Weedon teaches a computer system product which employs a storage medium for causing a computer to function as a transaction processing system for processing a

plurality of transactions in parallel with respect to data, the computer program product comprising:

a first computer program code loaded in a processor for causing the computer to produce a copy *(i.e. The invention permits users to enjoy both a performance benefit from the exclusive mode caching as well as a concurrency benefit of having multiple copies of the object accessible at one time, col. 3, lines 1-11)* of the data at a time of starting an access to the data by each transaction *(i.e. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary)*;

a second computer program code loaded in a processor for causing the computer to judge whether *(i.e. On the other hand, if the test at step 280 determines that the counter is zero, then the just completed transaction is the last outstanding transaction. In that case, a test is made at step 235 to determine whether the transaction is to be committed to the database. As described above, if the transaction is to be rolled back, then the process flow ends at step 240. Otherwise, the updates are committed to the data store by copying the record in that transaction to the data store, as shown at step 245, and by putting the contents of that object in the exclusive mode cache 150, as indicated at step 250, col. 5, lines 57-67)* a collision between one of reading access or writing access *(i.e. completing any current transactions by selectively committing any updates that were made to the object with a write operation to the data store, Summary)* to be made by a first transaction with respect to a copy of the data for the first transaction *(i.e. responding to a first transaction by retrieving a cache version of the object from the exclusive access cache, Summary)* and another one of the reading access or writing access made by the second transaction with respect to a copy of the data for the second transaction will occur or not *(i.e. the*

last transaction being either the first transaction or one of the concurrent transactions, if any, Summary) when the first transaction and the second transaction are accepted at the same time as concurrent transaction for accessing the same location of the data *(i.e. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary)*;

a third computer program code loaded in a processor for causing the computer to carry out a processing *(i.e. If the counter is not zero, that indicates that concurrent transactions are being processed. Updates of the just-completed transaction (from step 260), if any, may have to be copied to the data store. A test is made at step 285 to determine whether there is any non-conflicting update to commit, and if there is, the data store is updated at step 290. In this circumstance, multiple copies of the data are out for processing, with the first transaction having exclusive access to the exclusive mode cache, and subsequent concurrent transactions having access to copies of the data from the data store. In sharp contrast to the prior art, these transactions can be updated to the data store while the exclusive mode cache is locked up for the first transaction, col. 5, lines 44-56)* for avoiding the collision due to the concurrent transactions when the judging step judges that the collision will occur *(i.e. if the test at step 280 determines that the counter is zero, then the just completed transaction is the last outstanding transaction. In that case, a test is made at step 235 to determine whether the transaction is to be committed to the database. As described above, if the transaction is to be rolled back, then the process flow ends at step 240. Otherwise, the updates are committed to the data store by copying the record in that transaction to the data store, as shown at step 245, and by putting the contents of that object in the exclusive mode cache 150, as indicated at step 250, col. 5, lines 57-67)*;

and a fourth computer program code loaded in a processor for causing the computer to reflect a writing access made by the first transaction with respect to a copy of the data for the first transaction *(i.e. As part of the completion process, the sharing counter is decremented. If the value after decrementing is zero, that indicates that the transaction currently completing is the only transaction to have a copy of the record at that point in time. If the post-decrement counter value is zero, and if the transaction is committing (meaning that the record is being copied back to the database) then this record is also put back into the exclusive mode cache, to be used by subsequent transactions, col. 6, lines 48-56)*, on the data, when the first transaction is to be finished normally, and reflecting the writing access also on a copy of the data for the second transaction if the second transaction is not finished yet *(i.e. Each transaction is subsequently completed, meaning that it is either committed or rolled back. If the transaction is committed, it means that the copy of the record is written back to the database (if it was modified). If the transaction was rolled back, it means that the copy of the record is thrown away, col. 6, lines 41-47).*

Weedon does not specifically teach hierarchical data limitation.

Schrader teaches this limitation *(i.e. The database module 1407 is an interface to the user's data, which is stored in the transaction database. Records may be fetched or saved by the database module 1407 and this module allows all the other modules to access the transaction database. The database module 1403 preferably stores the user's data in combined relational-hierarchical data model, though other data models may also be employed. A hierarchical model is used to organize accounts per financial institution, such that all transactions for each account are stored with the account. A relational model is used for individual transactions, to provide*

associations between payees, amounts, and individual transactions within each account. Data that is stored includes payment data, transactions data, funds transfer data, and e-mail data. Transaction records may be variable length. Each record in the transaction data is marked as being either in the online statement 150, the mini checkbook 181, or the out box 167, and as being either reconciled or unreconciled. This marking allows for subsequent auto-reconciliation and reporting. Some transactions in the online statement 150 may be unreconciled if the user did not enter them first in the mini checkbook 181, col. 13, line 65 to col. 14, line 19).

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon and Schrader at the time the invention was made to modify the system of Weedon to include the hierarchical data as taught by Schrader. One of ordinary skill in the art would be motivated to make this combination in order to store the user's data in combined relational-hierarchical data model in view of Schrader (col. 13, line 14 to col. 14, line 19), as doing so would give the added benefit of a hierarchical model is used to organize accounts per financial institution, such that all transactions for each account are stored with the account as taught by Schrader (col. 13, line 65 to col. 14, line 19).

As per claim 2, Weedon teaches the concurrency control method of claim 1, wherein the judging step whether the collision will occur or not, according to whether data looked up by making the reading access without taking the writing access into consideration and data looked up by making the reading access by taking the writing access into consideration are identical or not *(i.e. a cache version of the object from the exclusive access cache; b. responding to any*

concurrent transactions by retrieving respective new instances of the object from the data store, Summary).

As per claim 3, Weedon teaches the concurrency control method of claim 1, wherein when the first transaction is to make the reading access with respect to a copy of the hierarchical data, the judging step judges whether the collision will occur or not according to whether first data looked up by making the reading access with respect to a copy of the hierarchical data for the first transaction and second data looked up by making the reading access with respect to data obtained by merging a copy of the hierarchical data for the first transaction and a copy of the hierarchical data for the second transaction are identical or not *(i.e. a cache version of the object from the exclusive access cache; b. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary).*

As per claim 4, Weedon teaches the concurrency control method of claim 3, wherein the judging step judges that the collision will not occur when the first data and the second data are judged as identical for all transactions that can be the second transaction, and judges that the collision will occur otherwise *(i.e. a cache version of the object from the exclusive access cache; b. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary).*

As per claim 5, Weedon teaches the concurrency control method of claim 1, further comprising: making the writing access with respect to a shared copy produced by copying the

hierarchical data in order to reflect writing accesses made by all transactions that make accesses to the hierarchical data, when the first transaction is to make the writing access with respect to a copy of the hierarchical data *(i.e. completing any current transactions by selectively committing any updates that were made to the object with a write operation to the data store; and d. completing a last transaction by selectively committing any updates that were made to the object with a write to the data store and a deposit to the exclusive access cache, the last transaction being either the first transaction or one of the concurrent transactions, if any, Summary)*; wherein when the first transaction is to make the reading access with respect to a copy of the hierarchical data, the judging step judges whether the collision will occur or not according to whether first data looked up by making the reading access and second data looked up by making the reading access with respect to the shared copy of the hierarchical data are identical or not *(i.e. a cache version of the object from the exclusive access cache; b. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary).*

As per claim 6, Weedon teaches the concurrency control method of claim 5, wherein the judging step judges that the collision will not occur when the first data and the second data are judged as identical, and judges that the collision will occur when the first data and the second data are judged as not identical *(i.e. a cache version of the object from the exclusive access cache; b. responding to any concurrent transactions by retrieving respective new instances of the object from the data store, Summary).*

As per claim 8, Weedon teaches the concurrency control method of claim 1, wherein

when the first transaction is to make the writing access with respect to a copy of the hierarchical data, the judging step judges whether the collision will occur or not according to whether first data looked up by making the reading access of the second transaction and second data looked up by making the reading access of the second transaction with respect to a state of the hierarchical data after the writing access are identical or not *(i.e. If the counter is not zero, that indicates that concurrent transactions are being processed. Updates of the just-completed transaction (from step 260), if any, may have to be copied to the data store. A test is made at step 285 to determine whether there is any non-conflicting update to commit, and if there is, the data store is updated at step 290, col. 5, lines 44-56)*, for all reading accesses by all transactions that make accesses to the hierarchical data and that can be the second transaction *(i.e. Since the exclusive mode cache is in use by the first transaction, the concurrent transactions each receive respective copies of the record from the data store, and the counter is used to index the number of shared copies that are outstanding and being operated upon at any given moment, col. 6, lines 1-24).*

As per claim 9, Weedon teaches the concurrency control method of claim 8, wherein the judging step judges that the collision will not occur when the first data and the second data are judged as identical for all reading accesses of all transactions that make accesses to the hierarchical data and that can be the second transaction, and judges that the collision will occur otherwise *(i.e. If the counter is not zero, that indicates that concurrent transactions are being processed. Updates of the just-completed transaction (from step 260), if any, may have to be copied to the data store. A test is made at step 285 to determine whether there is any non-*

conflicting update to commit, and if there is, the data store is updated at step 290, col. 5, lines 44-56).

As per claim 15, Weedon teaches the concurrency control method of claim 8, wherein the judging step obtains the second data as data obtained by making the writing access of the second transaction with respect to a state after the writing access was made with respect to a copy of the hierarchical data for the first transaction *(i.e. If the counter is not zero, that indicates that concurrent transactions are being processed. Updates of the just-completed transaction (from step 260), if any, may have to be copied to the data store. A test is made at step 285 to determine whether there is any non-conflicting update to commit, and if there is, the data store is updated at step 290, col. 5, lines 44-56)*, and then making the reading access with respect to a state of the hierarchical data after the writing access of the second transaction *(i.e. Since the exclusive mode cache is in use by the first transaction, the concurrent transactions each receive respective copies of the record from the data store, and the counter is used to index the number of shared copies that are outstanding and being operated upon at any given moment, col. 6, lines 1-24).*

9. Claim 21 is rejected under 35 U.S.C. 103(a) as being unpatentable over Raz et al. (US Patent No. 5,504,899), in view of Schrader et al. (US Patent No. 5,903,881).

As per claim 21, Raz teaches computer product which employs a storage medium for causing a computer to function as a transactions processing system for processing a plurality of transactions in parallel with respect to data, the computer program product comprising:

a first computer program code loaded in a processor *(i.e. transaction processing system, col. 6, line 40-51)* for causing the computer to accept transactions which are temporarily overlapping *(i.e when a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions, col. 6, lines 40-51)*;

a second computer program code loaded in a processor for causing the computer to produce a copy *(i.e. a back-up copy, col. 9, line 52 to col. 10, line 5)* of the data at a time of starting *(i.e. before the results of a transaction are written to state memory, col. 9, line 52 to col. 10, line 5)* execution of each transaction *(i.e. Transaction processing is based upon the technique of making a back-up copy of state memory before the results of a transaction are written to state memory, and also writing in non-volatile memory an indication of either a first processing phase in which the back-up copy is being made, or a second processing phase in which the results of a transaction are being written to state memory, in order to indicate which copy might have been corrupted during a failure. For making a back-up copy of state memory, for example, the non-volatile memory 23 includes two banks of state memory 28 and 29. To provide an indication of which bank of state memory might have been corrupted by a failure, the non-volatile memory 23 includes a memory location 30 for storing a switch or flag, col. 9, line 52 to col. 10, line 5)*;

a third computer program code loaded in a processor for causing the computer to judge *(i.e. aborting or delaying commitment of transactions, col. 6, lines 19-39)* whether or not the copy of a first transaction will conflict with the copy of a second transaction after execution of

the first and second transaction *(i.e. The conflicts, for example, are indicated by a serializability graph, maintained in each resource manager, wherein nodes represent transactions, directed edges represent direct conflicts, and paths including more than one edge represent indirect conflicts. This method can be used with any other mechanism that ensures local serializability, without affecting that mechanism's resource access scheduling strategy. Therefore, the method of the present invention can be used with existing mechanisms for ensuring local serializability or with a mechanism that is selected or optimized for each processor or process according to the nature of the transactions, col. 6, lines 19-39)*;

a fourth computer program coded loaded in a processor for causing the computer, when it is judged that a conflict will occurs between the first and second transactions, to halt the execution of the second transaction *(i.e. aborting or delaying commitment of transactions, col. 6, lines 19-39)* until the execution of the first transaction has been completed *(i.e. when a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions, col. 6, lines 40-51)*; and

a fifth computer program code loaded in a processor for causing the computer to reflect the copy of the first transaction in the data, when the first transaction is finished normally, and reflect the copy of the first transaction in the copy of the second transaction, when the second transaction is not finished yet *(i.e. when a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the*

second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions, col. 6, lines 40-51).

Raz does not expressly teach hierarchical data limitation.

Schrader teaches this limitation *(i.e. The database module 1407 is an interface to the user's data, which is stored in the transaction database. Records may be fetched or saved by the database module 1407 and this module allows all the other modules to access the transaction database. The database module 1403 preferably stores the user's data in combined relational-hierarchical data model, though other data models may also be employed. A hierarchical model is used to organize accounts per financial institution, such that all transactions for each account are stored with the account. A relational model is used for individual transactions, to provide associations between payees, amounts, and individual transactions within each account. Data that is stored includes payment data, transactions data, funds transfer data, and e-mail data. Transaction records may be variable length. Each record in the transaction data is marked as being either in the online statement 150, the mini checkbook 181, or the out box 167, and as being either reconciled or unreconciled. This marking allows for subsequent auto-reconciliation and reporting. Some transactions in the online statement 150 may be unreconciled if the user did not enter them first in the mini checkbook 181, col. 13, line 65 to col. 14, line 19).*

It would have been obvious to one of ordinary skill of the art having the teaching of Raz and Schrader at the time the invention was made to modify the system of Raz to include the hierarchical data as taught by Schrader. One of ordinary skill in the art would be motivated to make this combination in order to store the user's data in combined relational-hierarchical data

model in view of Schrader (col. 13, line 14 to col. 14, line 19), as doing so would give the added benefit of obtaining a hierarchical model to organize accounts per financial institution, such that all transactions for each account are stored with the account as taught by Schrader (col. 13, line 65 to col. 14, line 19).

10. Claims 10-13, 16, 18 are rejected under 35 U.S.C. 103(a) as being unpatentable over Weedon et al (US Patent No. 6,732,239), in view of Schrader et al. (US Patent No. 5,903,881), and further in view of Raz et al. (US Patent No. 5,504,899).

As per claim 10, Weedon, Schrader teach the concurrency control method of claim 8, but do not teach recording an access sequence of accesses made with respect to a copy of the hierarchical data by each transaction, for each one of all transactions that make accesses to the hierarchical data; wherein the judging step obtains all reading accesses of all transactions that make accesses to the hierarchical data and that can be the second transaction, by looking up a record of the access sequence.

Raz teaches recording an access sequence *(i.e. a transaction identification code 31, 32 is written into the state memory along with the time 33, 34 at which the results of the transaction were first written (i.e., committed) to state memory, col. 10, lines 6-14)* of accesses made with respect to a copy of the hierarchical data by each transaction, for each one of all transactions that make accesses to the hierarchical data *(i.e. The conflicts, for example, are indicated by a serializability graph, maintained in each resource manager, wherein nodes represent transactions, directed edges represent direct conflicts, and paths including more than one edge represent indirect conflicts, col. 6, lines 19-39)*; wherein the judging step obtains all reading

accesses of all transactions that make accesses to the hierarchical data and that can be the second transaction, by looking up a record of the access sequence *(i.e. This method can be used with any other mechanism that ensures local serializability, without affecting that mechanism's resource access scheduling strategy. Therefore, the method of the present invention can be used with existing mechanisms for ensuring local serializability or with a mechanism that is selected or optimized for each processor or process according to the nature of the transactions, col. 6, lines 19-39).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz at the time the invention was made to modify the system of Weedon, Schrader to include the limitations as taught by Raz. One of ordinary skill in the art would be motivated to make this combination in order to have a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions in view of Raz (Summary), as doing so would give the added benefit of providing a computing system across distributed transactions over multiple resource managers by selectively committing global transactions and aborting or delaying commitment of transactions to enforce an order of commitment of global transactions that is the same as an order of conflicts among the global transactions, including indirect conflicts through local transactions as taught by Raz (Summary).

As per claim 11, Weedon, Schrader teach the concurrency control method of claim 8, but

do not teach recording data looked up by making the reading accesses; wherein the judging step obtains the first data by looking up a record of the data looked up.

Raz teaches recording data looked up by making the reading accesses *(i.e. a transaction identification code 31, 32 is written into the state memory along with the time 33, 34 at which the results of the transaction were first written (i.e., committed) to state memory, col. 10, lines 6-14)*; wherein the judging step obtains the first data by looking up a record of the data looked up *(i.e. The conflicts, for example, are indicated by a serializability graph, maintained in each resource manager, wherein nodes represent transactions, directed edges represent direct conflicts, and paths including more than one edge represent indirect conflicts, col. 6, lines 19-39).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz at the time the invention was made to modify the system of Weedon, Schrader to include the limitations as taught by Raz. One of ordinary skill in the art would be motivated to make this combination in order to have a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions in view of Raz (Summary), as doing so would give the added benefit of providing a computing system across distributed transactions over multiple resource managers by selectively committing global transactions and aborting or delaying commitment of transactions to enforce an order of commitment of global transactions that is the same as an order of conflicts among the global transactions, including indirect conflicts through local transactions as taught by Raz (Summary).

As per claim 12, Weedon, Schrader teach the concurrency control method of claim 8, but do not teach the judging step obtains the first data as data obtained by making the writing access that was made by the second transaction before the reading access, with respect to a state of the hierarchical data at a start of the second transaction, and then making the reading access with respect to a state of the hierarchical data after the writing access.

Raz teaches the judging step obtains the first data as data obtained by making the writing access that was made by the second transaction before the reading access, with respect to a state of the hierarchical data at a start of the second transaction, and then making the reading access with respect to a state of the hierarchical data after the writing access *(i.e. The conflicts, for example, are indicated by a serializability graph, maintained in each resource manager, wherein nodes represent transactions, directed edges represent direct conflicts, and paths including more than one edge represent indirect conflicts, col. 6, lines 19-39).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz at the time the invention was made to modify the system of Weedon, Schrader to include the limitations as taught by Raz. One of ordinary skill in the art would be motivated to make this combination in order to have a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions in view of Raz (Summary), as doing so would give the added benefit of a computing system across distributed transactions over multiple resource managers by selectively committing global transactions and aborting or delaying

commitment of transactions to enforce an order of commitment of global transactions that is the same as an order of conflicts among the global transactions, including indirect conflicts through local transactions as taught by Raz (Summary).

As per claim 13, Weedon, Schrader the concurrency control method of claim 8, but do not specifically teach the steps of:

making the writing access with respect to a shared copy produced by copying the hierarchical data in order to reflect writing accesses made by all transactions that make accesses to the hierarchical data, when the first transaction is to make the writing access with respect to a copy of the hierarchical data; and

storing states of the shared copy at timings at which the writing accesses were made by some of the transactions that make accesses to the hierarchical data;

wherein the judging step obtains the first data as data obtained by reproducing a state of the hierarchical data at a timing at which the reading access was made by selecting one of stored states of the shared copy which is close to the state of the hierarchical data at a timing at which the reading access was made and making the writing access that was made by the second transaction with respect to a selected state of the shared copy according to need, and then making the reading access with respect to a reproduced state of the hierarchical data.

Raz teaches:

making the writing access with respect to a shared copy produced by copying the hierarchical data in order to reflect writing accesses made by all transactions that make accesses to the hierarchical data, when the first transaction is to make the writing access with respect to a

copy of the hierarchical data *(i.e. Transaction processing is based upon the technique of making a back-up copy of state memory before the results of a transaction are written to state memory, and also writing in non-volatile memory an indication of either a first processing phase in which the back-up copy is being made, or a second processing phase in which the results of a transaction are being written to state memory, in order to indicate which copy might have been corrupted during a failure. For making a back-up copy of state memory, for example, the non-volatile memory 23 includes two banks of state memory 28 and 29. To provide an indication of which bank of state memory might have been corrupted by a failure, the non-volatile memory 23 includes a memory location 30 for storing a switch or flag, col. 9, line 52 to col. 10, line 5)*; and

storing states of the shared copy at timings at which the writing accesses were made by some of the transactions that make accesses to the hierarchical data *(i.e. a transaction identification code 31, 32 is written into the state memory along with the time 33, 34 at which the results of the transaction were first written (i.e., committed) to state memory, col. 10, lines 6-14)*;

wherein the judging step obtains the first data as data obtained by reproducing a state of the hierarchical data at a timing at which the reading access was made by selecting one of stored states of the shared copy which is close to the state of the hierarchical data at a timing at which the reading access was made and making the writing access that was made by the second transaction with respect to a selected state of the shared copy according to need, and then making the reading access with respect to a reproduced state of the hierarchical data *(i.e. At the time that presence of a conflict is detected, as further described below with reference to FIG. 14, the order of performance is recorded in the global transaction commitment order serializability graph. The pertinent data in the graph of FIG. 8 and transactions list 93 is presented in pictorial form*

in FIG. 9. The flags that are set in the data structure of FIG. 8 correspond to edges 131 in the pictorial representation of FIG. 9. The direction of an edge 131 indicates the order of performance of the conflicting operations in the transactions. Once this order of performance is established, a corresponding global transaction commitment order is enforced by delaying commitment of transactions, or aborting transactions, col. 20, lines 35-54).

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz at the time the invention was made to modify the system of Weedon, Schrader to include the limitations as taught by Raz. One of ordinary skill in the art would be motivated to make this combination in order to have a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions in view of Raz (Summary), as doing so would give the added benefit of providing a computing system across distributed transactions over multiple resource managers by selectively committing global transactions and aborting or delaying commitment of transactions to enforce an order of commitment of global transactions that is the same as an order of conflicts among the global transactions, including indirect conflicts through local transactions as taught by Raz (Summary).

As per claim 16, Weedon, Schrader teach the concurrency control method of claim 8, but do not teach making the writing access with respect to a shared copy produced by copying the hierarchical data in order to reflect writing accesses made by all transactions that make accesses

to the hierarchical data, when the first transaction is to make the writing access with respect to a copy of the hierarchical data; and

storing states of the shared copy at timings at which the writing accesses were made by some of the transactions that make accesses to the hierarchical data;

wherein the judging step obtains the second data as data obtained by reproducing a state of the hierarchical data at a timing at which the reading access is to be made by selecting one of stored states of the shared copy which is close to the state of the hierarchical data at a timing at which the reading access is to be made, making the writing access that was made by the first transaction after that timing, with respect to a selected state of the shared copy, and making the writing access that was made by the second transaction according to need, and then making the reading access with respect to a reproduced state of the hierarchical data.

Raz teaches:

making the writing access with respect to a shared copy produced by copying the hierarchical data in order to reflect writing accesses made by all transactions that make accesses to the hierarchical data, when the first transaction is to make the writing access with respect to a copy of the hierarchical data *(i.e. Transaction processing is based upon the technique of making a back-up copy of state memory before the results of a transaction are written to state memory, and also writing in non-volatile memory an indication of either a first processing phase in which the back-up copy is being made, or a second processing phase in which the results of a transaction are being written to state memory, in order to indicate which copy might have been corrupted during a failure. For making a back-up copy of state memory, for example, the non-volatile memory 23 includes two banks of state memory 28 and 29. To provide an indication of*

which bank of state memory might have been corrupted by a failure, the non-volatile memory 23 includes a memory location 30 for storing a switch or flag, col. 9, line 52 to col. 10, line 5); and

storing states of the shared copy at timings at which the writing accesses were made by some of the transactions that make accesses to the hierarchical data *(i.e. a transaction identification code 31, 32 is written into the state memory along with the time 33, 34 at which the results of the transaction were first written (i.e., committed) to state memory, col. 10, lines 6-14)*;

wherein the judging step obtains the second data as data obtained by reproducing a state of the hierarchical data at a timing at which the reading access is to be made by selecting one of stored states of the shared copy which is close to the state of the hierarchical data at a timing at which the reading access is to be made, making the writing access that was made by the first transaction after that timing, with respect to a selected state of the shared copy, and making the writing access that was made by the second transaction according to need, and then making the reading access with respect to a reproduced state of the hierarchical data *(i.e. At the time that presence of a conflict is detected, as further described below with reference to FIG. 14, the order of performance is recorded in the global transaction commitment order serializability graph. The pertinent data in the graph of FIG. 8 and transactions list 93 is presented in pictorial form in FIG. 9. The flags that are set in the data structure of FIG. 8 correspond to edges 131 in the pictorial representation of FIG. 9. The direction of an edge 131 indicates the order of performance of the conflicting operations in the transactions. Once this order of performance is established, a corresponding global transaction commitment order is enforced by delaying commitment of transactions, or aborting transactions, col. 20, lines 35-54).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz at the time the invention was made to modify the system of Weedon, Schrader to include the limitations as taught by Raz. One of ordinary skill in the art would be motivated to make this combination in order to have a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions in view of Raz (Summary), as doing so would give the added benefit of providing a computing system across distributed transactions over multiple resource managers by selectively committing global transactions and aborting or delaying commitment of transactions to enforce an order of commitment of global transactions that is the same as an order of conflicts among the global transactions, including indirect conflicts through local transactions as taught by Raz (Summary).

As per claim 18, Weedon, Schrader teach the concurrency control method of claim 1, but do not explicitly teach when the judging step judges that the collision will occur, the carrying out step carries out the processing for keeping those transactions that are determined according to prescribed criteria among transactions related to the collision, to wait until other transactions related to the collision are finished.

Raz teaches the judging step judges that the collision will occur, the carrying out step *(i.e. For any two conflicting operations p.sub.1 [x], q.sub.2 [x] of any transactions T.sub.1, T.sub.2 respectively, ts(T.sub.1)<ts(T.sub.2) implies p.sub.1 [x]<q.sub.2 [x]. (Note: This Blocking*

Timestamp Ordering rule requires that conflicting operations are scheduled according to the timestamps order regardless of whether the respective transactions are committed.), col. 58, lines 29-32) carries out the processing for keeping those transactions that are determined according to prescribed criteria among transactions related to the collision, to wait until other transactions related to the collision are finished *(i.e. The conflicts, for example, are indicated by a serializability graph, maintained in each resource manager, wherein nodes represent transactions, directed edges represent direct conflicts, and paths including more than one edge represent indirect conflicts. This method can be used with any other mechanism that ensures local serializability, without affecting that mechanism's resource access scheduling strategy. Therefore, the method of the present invention can be used with existing mechanisms for ensuring local serializability or with a mechanism that is selected or optimized for each processor or process according to the nature of the transactions, col. 6, lines 19-39).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz at the time the invention was made to modify the system of Weedon, Schrader to include the limitations as taught by Raz. One of ordinary skill in the art would be motivated to make this combination in order to have a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions in view of Raz (Summary), as doing so would give the added benefit of providing a computing system across distributed transactions over multiple resource managers by selectively committing global transactions and aborting or

delaying commitment of transactions to enforce an order of commitment of global transactions that is the same as an order of conflicts among the global transactions, including indirect conflicts through local transactions as taught by Raz (Summary).

11. Claims 7, 14, 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Weedon et al (US Patent No. 6,732,239), in view of Schrader et al. (US Patent No. 5,903,881), and further in view of Peir et al. (US Patent No. 6,725,341).

As per claim 7, Weedon, Schrader teach the concurrency control method of claim 5, but do not teach wherein when there is an upper to a number of shared copies that can be recorded, those shared copies which have a higher possibility of being utilized at a time of reproducing a state in which the reading access is to be made later on are recorded at a higher priority, among the shared copies corresponding to states at times of the writing accesses with respect to the hierarchical data.

Raz teaches there is a number of shared copies that can be recorded, those shared copies which have a higher possibility of being utilized at a time of reproducing a state in which the reading access is to be made later on are recorded at a higher priority, among the shared copies corresponding to states at times of the writing accesses with respect to the hierarchical data *(i.e. permits the construction of a transaction processing system in which a second global transaction may read data written by a write operation of a first global transaction before the first global transaction is committed. In this case, depending on the respective order in which the two conflicting operations occur, either of the two global transactions may be aborted to ensure that*

the order of commitment is the same as the conflict order of the global transactions, col. 6, line 52 to col. 7, line 2).

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz at the time the invention was made to modify the system of Weedon, Schrader to include the limitations as taught by Raz. One of ordinary skill in the art would be motivated to make this combination in order to have a second global transaction performs a read operation before a conflicting write operation of a first global transaction is committed at a time when the second global transaction has not yet committed, the second global transaction is aborted to ensure that the order in which the global transactions are committed is not different from the conflict order of the global transactions in view of Raz (Summary), as doing so would give the added benefit of a computing system across distributed transactions over multiple resource managers by selectively committing global transactions and aborting or delaying commitment of transactions to enforce an order of commitment of global transactions that is the same as an order of conflicts among the global transactions, including indirect conflicts through local transactions as taught by Raz (Summary).

Weedon, Schrader, Raz do not teach an upper limit.

Peir teaches an upper limit *(i.e. Such rules that restrict memory write backs to lines likely to be needed in other caches serve to further limit the number of transfers over the system memory bus to alleviate risk of flooding the bus with L1 cache write-back activity, col. 5, lines 7-16).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz, Peir at the time the invention was made to modify the system of

Weedon, Schrader, Raz to include the limitations as taught by Peir. One of ordinary skill in the art would be motivated to make this combination in order to further limit the number of transfers over the system memory bus to alleviate risk of flooding the bus with L1 cache write-back activity in view of Peir (col. 5, lines 7-16), as doing so would give the added benefit of reducing the impact of cache line invalidation on cache efficiency in distributed cache multiprocessor computer systems as taught by Peir (col. 2, line 63 to col. 3, line 2).

As per claim 14, Raz teaches the concurrency control method of claim 13, wherein when there is a number of shared copies that can be recorded, those shared copies which have a higher possibility of being utilized at a time of reproducing a state in which the reading access is to be made later on are recorded at a higher priority, among the shared copies corresponding to states at times of the writing accesses with respect to the hierarchical data *(i.e. permits the construction of a transaction processing system in which a second global transaction may read data written by a write operation of a first global transaction before the first global transaction is committed. In this case, depending on the respective order in which the two conflicting operations occur, either of the two global transactions may be aborted to ensure that the order of commitment is the same as the conflict order of the global transactions, col. 6, line 52 to col. 7, line 2).*

Weedon, Schrader, Raz do not teach an upper limit.

Peir teaches an upper limit *(i.e. Such rules that restrict memory write backs to lines likely to be needed in other caches serve to further limit the number of transfers over the system memory bus to alleviate risk of flooding the bus with L1 cache write-back activity, col. 5, lines 7-16).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz, Peir at the time the invention was made to modify the system of Weedon, Schrader, Raz to include the limitations as taught by Peir. One of ordinary skill in the art would be motivated to make this combination in order to further limit the number of transfers over the system memory bus to alleviate risk of flooding the bus with L1 cache write-back activity in view of Peir (col. 5, lines 7-16), as doing so would give the added benefit of reducing the impact of cache line invalidation on cache efficiency in distributed cache multiprocessor computer systems as taught by Peir (col. 2, line 63 to col. 3, line 2).

As per claim 17, Raz teaches the concurrency control method of claim 16, wherein when there is a number of shared copies that can be recorded, those shared copies which have a higher possibility of being utilized at a time of reproducing a state in which the reading access is to be made later on are recorded at a higher priority, among the shared copies corresponding to states at times of the writing accesses with respect to the hierarchical data.

Weedon, Schrader, Raz do not teach an upper limit.

Peir teaches an upper limit *(i.e. Such rules that restrict memory write backs to lines likely to be needed in other caches serve to further limit the number of transfers over the system memory bus to alleviate risk of flooding the bus with L1 cache write-back activity, col. 5, lines 7-16).*

It would have been obvious to one of ordinary skill of the art having the teaching of Weedon, Schrader, Raz, Peir at the time the invention was made to modify the system of Weedon, Schrader, Raz to include the limitations as taught by Peir. One of ordinary skill in the

art would be motivated to make this combination in order to further limit the number of transfers over the system memory bus to alleviate risk of flooding the bus with L1 cache write-back activity in view of Peir (col. 5, lines 7-16), as doing so would give the added benefit of reducing the impact of cache line invalidation on cache efficiency in distributed cache multiprocessor computer systems as taught by Peir (col. 2, line 63 to col. 3, line 2).

Response to Arguments

12. Applicant's arguments regarding Iline does not suggest the features of claims 1-20 have been considered but are moot in view of the new ground(s) of rejection.

Conclusion

13. The prior art made of record and not relied upon is considered pertinent to applicant's disclosure.

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Miranda Le whose telephone number is (571) 272-4112. The examiner can normally be reached on Monday through Friday from 8:30 AM to 5:00 PM.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, John R. Cottingham, can be reached on (571) 272-7079. The fax number to this Art Unit is (571)-273-8300.

Any inquiry of a general nature or relating to the status of this application should be directed to the Group receptionist whose telephone number is (571) 272-2100.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free).

Mirandale

Miranda Le
January 03, 2008